Received May 16th, 2026; accepted May 30th, 2026. Date of publication June 30th, 2026
Digital Object Identifier: https://doi.org/10/25047/jtit.v13i1.494

Deep Learning-Based Tomato Leaf Disease Classification Using CNN, EfficientNetB0, and InceptionResNetV2

RIFQI AJI WIDARSO[1], ADI SUCIPTO[1], DHONY MANGGALA PUTRA[1], TAMARA MAHARANI[2]

[1]Department of Information Technology, Politeknik Negeri, Jember, Indonesia
[2]Akademi Komunitas Negeri Pacitan, Indonesia

CORESPONDING AUTHOR: RIFQI AJI WIDARSO (rifqiaji_w@polije.ac.id)

ABSTRACT Tomato leaf diseases threaten agricultural productivity because symptoms such as early blight, late blight, leaf mold, septoria leaf spot, and yellow curl virus often produce visually similar color changes, necrotic lesions, and leaf deformation. Manual visual diagnosis is subjective and depends heavily on expert experience; therefore, image-based deep learning is a relevant approach for supporting preliminary disease identification. This study compares five deep learning architectures, namely a custom convolutional neural network, EfficientNetB0, MobileNetV2, DenseNet121, and InceptionResNetV2, for classifying six tomato leaf categories using 7,192 images from a Kaggle dataset. These architectures were selected to represent a spectrum ranging from training from scratch (custom CNN) to lightweight transfer learning models (EfficientNetB0 and MobileNetV2) and more complex deep architectures (DenseNet121 and InceptionResNetV2), thereby enabling a systematic evaluation of the relationship between model complexity and classification performance on a medium-sized agricultural image dataset. The research workflow includes dataset preparation, image resizing and normalization, model training using the Adam optimizer, and evaluation through accuracy, loss, precision, recall, F1-score, and confusion matrix analysis EfficientNetB0,MobileNetV2, and DenseNet121 were initialized with ImageNet pretrained weights and evaluated on a validation split, the custom CNN and InceptionResNetV2 were evaluated on a held-out test split. This distinction in evaluation source is an acknowledged limitation and is discussed fully in Section III. EfficientNetB0 achieved the best validation accuracy of 89.44% after 20 epochs, followed by MobileNetV2 at 85.12%, DenseNet121 at 82.77%, the custom CNN at 70.69% test accuracy, and InceptionResNetV2 at 45.76% test accuracy. The results indicate that lightweight transfer learning models are more suitable for medium-sized agricultural image datasets than large architectures trained for only a few epochs. Future work should validate the model using real field images, harmonize all models on the same test set, and report class-wise metrics to ensure reliability before deployment as a farmer-oriented diagnostic support system.

KEYWORDS: Convolutional neural network, Deep learning, Disease classification, EfficientNetB0, Tomato leaf

I.INTRODUCTION

Tomato (Solanum lycopersicum) ranks among the most commercially significant horticultural crops worldwide, contributing substantially to food security, rural income, and agro-industrial supply chains. Despite its economic importance, tomato cultivation is highly susceptible to a range of foliar diseases. Early blight, caused by Alternaria solani, produces concentric necrotic rings on older leaves. Late blight, attributed to Phytophthora infestans, rapidly spreads under cool and humid conditions, generating water-soaked lesions that collapse leaf tissue. Leaf mold, induced by Passalora fulva, colonizes the lower leaf surface with velvety olive-green patches. Septoria leaf spot, caused by Septoria lycopersici, forms numerous small circular lesions with dark borders. Yellow leaf curl virus, transmitted by whiteflies, causes leaf curling, interveinal chlorosis, and stunted growth. These diseases share overlapping symptom spectra necrotic tissue, chlorotic margins, and structural deformation making visual differentiation unreliable without laboratory support[1],[2].

Conventional disease management relies on field scouting by trained agronomists, which is time-consuming, costly, and prone to inter-observer variability[3]. Remote diagnostic services are largely inaccessible to smallholder farmers in developing regions. Several recent studies have documented the limitations of traditional inspection

and highlighted the pressing need for automated, data-driven approaches that can support preliminary disease identification at the point of need. Image-based deep learning has emerged as one of the most promising directions in this regard because it eliminates the requirement for manual feature engineering: hierarchical visual representations are learned end-to-end from labeled examples.

Convolutional neural networks (CNNs) are particularly well suited for image classification tasks. They apply learnable filters across spatial dimensions, progressively extracting low-level features such as edges and color gradients in early layers, and high-level disease-specific patterns lesion shape, texture, and color abnormality in deeper layers[4]. Transfer learning extends this capability by initializing a model with weights pretrained on large general-purpose image corpora such as ImageNet, then fine-tuning the final layers on domain-specific data. This strategy substantially reduces the volume of labeled agricultural images required for convergence and often yields superior accuracy relative to training from scratch.

Prior work has evaluated a variety of architectures for tomato leaf disease detection but with notable methodological limitations. CNN configurations for tomato leaf disease and found that deeper architectures consistently outperformed shallower ones. However, all models were trained from scratch, leaving the contribution of pretrained initialization unquantified[]. CNN classification to a local tomato leaf dataset and reported that accuracy varied significantly depending on preprocessing and augmentation strategies, without controlling for architecture[]. Inception-V3 with the Adam optimizer has demonstrated high accuracy under controlled laboratory conditions. Standard CNNs have been applied to multi-class leaf datasets with reasonable performance. VGG-19 provides deeper feature extraction through its uniform 3×3 convolutional stack. MobileNetV2 and DenseNet have been explored as lightweight alternatives that reduce computational cost while maintaining competitive accuracy. Android-based deployment studies further underscore the practical necessity of architectures that remain accurate under field constraints variable lighting, leaf occlusion, and non-uniform camera quality. Recent benchmark surveys covering tomato, rice, maize, chili, and coffee diseases consistently find that lightweight transfer learning models such as EfficientNet and MobileNet variants perform favourably on moderate-sized datasets[5],[6],[7].

EfficientNet-based transfer learning to three tomato leaf datasets of increasing size and reported overall classification accuracies of 97.3%, 99.2%, and 99.5% for 3,000, 8,000, and 10,000 images respectively, demonstrating a clear correlation between dataset scale and model performance[]. A lightweight T-Net integrating multiple CNN backbones for tomato disease classification and showed that architectural customization can improve generalization beyond off-the-shelf transfer learning.Ensemble models combining DenseNet121 MobileNetV2 and ResNet variants for rice disease detection have reported accuracies above 99%, highlighting the complementary nature of these architectures, MobileNetV2 for tomato disease classification and reported competitive accuracy with faster inference time on constrained devices[8].

Despite this growing body of research, three important gaps remain. First, most studies evaluate a single model or compare architectures trained on different datasets with inconsistent hyperparameters, making reliable cross architecture conclusions difficult. Second, the effect of a severely limited training budget on large-parameter models such as InceptionResNetV2 when applied to medium sized agricultural datasets has not been systematically examined. Third, no published study has simultaneously placed a training-fromscratch baseline, lightweight transfer learning models, and large multi-scale architectures under documented evaluation conditions on the same balanced six-class tomato leaf disease dataset.

This article addresses these gaps by comparing five architectures custom CNN, EfficientNetB0, MobileNetV2, DenseNet121, and InceptionResNetV2 on the same six-class tomato leaf dataset. The primary contribution of this study lies in providing a structured comparative evaluation of multiple deep learning architectures using a nearly balanced six-class tomato leaf disease dataset. The rationale for selecting these specific architectures is as follows. The custom CNN serves as an interpretable baseline that quantifies the performance gain attributable to transfer learning. EfficientNetB0 was chosen because its compound scaling methodology balances model depth, width, and resolution in a principled manner, and it has consistently ranked among the top performers on agricultural image benchmarks InceptionResNetV2 was included to investigate whether a large-parameter model can match compact models when the training budget is constrained a question not addressed in prior tomato leaf disease studies. MobileNetV2 and DenseNet121 complete the spectrum as widely used lightweight and densely connected alternatives in agricultural classification research The primary novelty of this study lies in placing all five architectures under documented experimental conditions on a balanced six-class tomato leaf disease dataset while simultaneously analyzing the impact of training duration on large-scale models.

In addition, this research presents a comprehensive mathematical formulation covering convolutional neural network operations, optimization processes, and evaluation metrics to improve methodological clarity and reproducibility.

The study also delivers an in-depth analysis of several critical factors affecting model performance, including the number of training epochs, input image resolution, and consistency of evaluation procedures.Furthermore,practical recommendations are proposed for the development of farmer-oriented diagnostic systems that can be efficiently deployed on mobile and edge computing devices in real agricultural environments [9].

II.METHOD

2.1 Model Architecture Review

Figure 1 illustrates the overall research methodology employed in this study for tomato leaf disease classification using deep learning. The pipeline begins with the collection of tomato leaf images in RGB color space (.jpg/.png format), which are then subjected to a preprocessing stage involving resizing, normalization, and data augmentation. Preprocessing ensures that all input images conform to a consistent spatial resolution and pixel value range, while data augmentation techniques such as flipping, rotation, and brightness adjustment are applied to artificially expand the training set and improve the model's generalization capability.

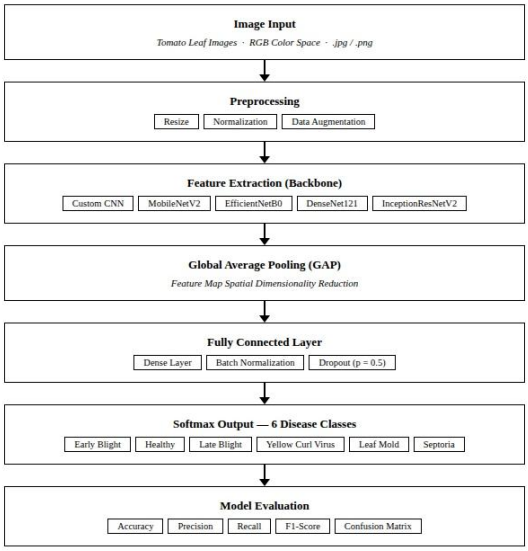

FIGURE 1. Methodology for tomato disease classification

The core of the methodology lies in the feature extraction stage, where five distinct backbone architectures are evaluated: Custom CNN, MobileNetV2, EfficientNetB0, DenseNet121, and InceptionResNetV2. Each backbone serves as an independent feature extractor, capturing hierarchical visual patterns from the input images. The extracted feature maps are subsequently passed through a Global Average Pooling (GAP) layer, which reduces the spatial dimensions of each feature map to a single scalar value per channel, effectively compressing high-dimensional representations while mitigating the risk of overfitting. The resulting feature vector is then fed into a fully connected classification head consisting of a dense layer, batch normalization, and a dropout layer with a rate of 0.5 to further regularize the model during training.

Finally, the classification head terminates with a softmax activation function that produces a probability distribution across six disease classes: Early Blight, Healthy, Late Blight, Yellow Curl Virus, Leaf Mold, and Septoria. The predicted class corresponds to the category with the highest probability score. To rigorously assess the performance of each backbone architecture, the model is evaluated using five standard metrics — Accuracy, Precision, Recall, F1-Score, and Confusion Matrix — enabling a comprehensive and comparative analysis of each model's ability to correctly identify and distinguish between the six tomato leaf disease categories.

2.1.1 Custom Convolutional Neural Network

A convolutional neural network is a hierarchical architecture purpose-built for grid-structured data such as digital images[10]. The fundamental operation is discrete 2-D convolution: a learned kernel slides across an input tensor and computes weighted sums at each spatial position, progressively building feature representations that are increasingly abstract and spatially compact. CNNs learn this hierarchy automatically through backpropagation, relieving practitioners of manual feature engineering.

The custom CNN designed in this study is a purpose-built architecture trained entirely from randomly initialized weights, without any pretrained backbone. This design choice serves a specific methodological function: it establishes an interpretable baseline that quantifies how much performance is attributable purely to the network's ability to learn disease-discriminative features from the available training data, without the prior knowledge encoded in ImageNet-pretrained weights.

The architecture consists of four sequential convolutional blocks. Each block contains a 3 × 3 Conv2D layer followed by batch normalization, a Rectified Linear Unit (ReLU) activation function, and a 2 × 2 max pooling layer. The number of filters increases progressively across blocks 32, 64, 128, and 256 filters respectively so that deeper layers can represent more complex and abstract disease patterns. Batch normalization is applied after each convolution to stabilize the internal feature distribution, accelerate convergence, and reduce sensitivity to weight initialization. Max pooling reduces spatial dimensions by half at each block, progressively increasing the model's spatial invariance to small translations in lesion position.

After the fourth convolutional block, a global average pooling (GAP) layer collapses each feature map to a single scalar by computing the spatial average across all positions. GAP serves as a regularizer that reduces overfitting compared to a traditional flatten layer, because it forces the network to produce activations that are globally representative of each class rather than memorizing spatially specific patterns. The resulting feature vector is passed through three fully connected (dense) layers. A dropout layer with rate p = 0.5 is inserted between the dense layers to further mitigate overfitting by randomly deactivating half of the neurons during each training step. The final layer applies a six-class softmax activation to produce a probability distribution over the disease categories.

The model is compiled with the Adam optimizer, a learning rate of 0.0001, and categorical cross-entropy loss. Training was conducted for 20 epochs with early stopping monitoring validation accuracy, using a batch size of 32. The complete trainable parameter count is approximately 2.1 million, making it substantially more compact than the transfer learning architectures compared in this study.

The custom CNN in this study comprises four convolutional blocks, each containing a 3×3 Conv2D layer, batch normalization, ReLU activation, and 2×2 max pooling. After the fourth block, a global average pooling layer reduces the spatial dimensions to a single feature vector, which is then passed through three fully connected layers with dropout (rate = 0.5) before the six-class softmax output[11].

Custom CNNs trained without pretrained weights offer full interpretability and architectural control. Their main limitation is data hunger: the model must learn both low-level visual primitives and high-level disease semantics from the training set alone. With approximately 7,200 images, a four-block CNN can converge to a reasonable baseline but typically cannot match the representational richness of transfer-learned models[4] .

2.1.2 MobileNetV2

MobileNetV2 is a family of lightweight neural architectures developed specifically for mobile and resource-constrained inference environments. Its design principle centers on inverted residual blocks combined with linear bottlenecks. Each inverted residual block first expands the number of channels using a 1×1 pointwise convolution, then applies a depthwise separable convolution within the expanded feature space, and finally projects back to a lower-dimensional representation using another 1×1 convolution. Decoupling depthwise and pointwise operations reduces the multiply-accumulate count by a factor of roughly 8–9 relative to a standard 3×3 convolution of equal channel depth, enabling real-time inference on smartphones and embedded systems.

Recent studies confirm MobileNetV2's suitability for agricultural disease detection. A precision farming benchmark fusing MobileNetV2 and EfficientNetB0 achieved 89.5% global accuracy on a 22-class crop disease dataset encompassing cashew, cassava, maize, and tomato, with inference below 10 ms per image following post-training quantization [12]. A feature-concatenation approach combining MobileNetV2 and NASNetMobile for tomato disease classification attained 95% accuracy [13], further validating the architecture's utility for foliar pathology tasks.

2.1.3 EfficientNetB0

EfficientNet employs a compound scaling methodology that simultaneously increases network depth, width, and input resolution under a fixed computational budget [5]. Rather than scaling any single dimension in isolation as earlier networks such as ResNet (depth) or WideResNet (width) do compound scaling applies a uniform coefficient derived from a neural architecture search baseline. EfficientNetB0 is the smallest member of the family, containing approximately 5.3 million parameters and using MBConv blocks equipped with squeeze-and-excitation modules. These modules compute global average-pooled channel statistics and apply learned multiplicative weights to recalibrate channel responses adaptively, allowing the network to emphasize informative disease-related features while suppressing irrelevant background texture.

EfficientNetB0 has been ranked among the highest-performing architectures for plant disease classification . When combined with convolutional block attention modules (CBAM), it achieved 86.89% classification accuracy on the DiaMOS plant disease dataset, outperforming standalone EfficientNetB0 [6]. On larger tomato datasets with 8,000 to 10,000 images, pure EfficientNet transfer learning surpassed 99% accuracy [7]. The architecture's favorable accuracy-to-parameter ratio makes it a natural candidate for edge-deployable diagnostic tools.

2.1.4 DenseNet121

DenseNet introduces dense connectivity: every layer within a dense block receives direct concatenated feature maps from all preceding layers[14]. This architecture simultaneously addresses three problems common in deep networks: vanishing gradients are mitigated because each layer has a direct gradient path to the loss; feature propagation is strengthened because early low-level features remain accessible throughout the network; and parameter efficiency is improved because layers do not need to re-learn representations already

captured at shallower levels. DenseNet121 consists of four dense blocks interleaved with transition layers that compress feature map dimensions.

For agricultural image classification, DenseNet121 is particularly relevant because disease-related patterns early lesion discoloration, mold colony boundaries, viral mosaic textures appear at multiple scales of visual abstraction. A recent ensemble study reported DenseNet121 achieving 99.81% classification accuracy on the PlantVillage dataset when fine-tuned with appropriate learning rate scheduling. LeafDoc-Net, which concatenates DenseNet121 and MobileNetV2 with attention-enhanced transition layers, demonstrated superior accuracy on cassava and wheat leaf disease datasets relative to either backbone alone [13].

2.1.5 InceptionResNetV2

InceptionResNetV2 merges two foundational design philosophies: Inception modules, which capture multi-scale spatial features by running parallel convolutional branches of different kernel sizes, and residual connections, which preserve gradient flow across many layers and enable much deeper networks to train stably [15]. The architecture contains approximately 55.3 million parameters, making it one of the largest models evaluated in this study. Its multi-scale receptive fields are theoretically advantageous for detecting lesions at varying sizes across the leaf surface.

In practice, however, large architectures require proportionally more training data, longer training schedules, and careful preprocessing to converge on the classification head. When the backbone is initialized with ImageNet weights and only the head is fine-tuned for a limited number of epochs on a medium-sized agricultural dataset, the base feature extractor may remain misaligned with the target domain, resulting in underfitting. This behavior has been documented in comparative studies of rice and maize disease detection, where smaller EfficientNet and MobileNet variants outperformed larger Inception-based architectures under equivalent training budgets[15].

2.2 Dataset

This study uses the six-class Tomato Leaf Disease Dataset from Kaggle: *(kaggle.com/syedhashirali260/tomato-leaf-disease-dataset-6-classes)*. The local dataset contains 7,192 images and is nearly balanced across the six classes, as summarized in Table 1 and illustrated in Figure 1.

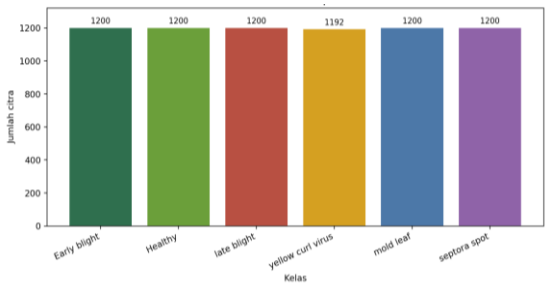

FIGURE 2. Distribution of tomato leaf images in the local folder

The class distribution, summarized in Table 1, is nearly balanced, with five classes containing exactly 1,200 images and the yellow curl virus class containing 1,192. This near-balance is a deliberate selection criterion: it reduces the risk that accuracy metrics are inflated by class frequency bias and allows model evaluation to reflect per-class discriminative performance more transparently.

Image quality across the dataset is heterogeneous: some images were captured under controlled indoor lighting on uniform backgrounds, while others appear to originate from field photography with natural illumination, leaf occlusion, and complex backgrounds. This variability provides a moderate degree of domain diversity, though it falls short of the diversity present in dedicated real-field datasets such as those used in the Frontiers in Plant Science benchmark.

TABLE 1. Dataset Distribution in the Local Folder

Class	Number of images
Tomato_Early_blight	1,200
Tomato_Healthy	1,200
Tomato_leaf_late_blight	1,200
Tomato_leaf_yellow_curl_virus	1,192
Tomato_mold_leaf	1,200
Tomato_septora_leaf_spot	1,200
Total local	7,192

2.3 Experimental Configuration

Five independent experimental implementations in Jupyter Notebook environments served as the primary implementation sources. Each Jupyter Notebook environtment differs in framework, input resolution, data split, and training duration, reflecting the heterogeneous practices common in the deep learning literature.

The custom CNN Jupyter Notebook environtment was implemented in PyTorch. Input images were resized to 224×224 pixels and normalized using per-channel mean and standard deviation values of 0.5. A batch size of 32 was used with a 70:20:10 training, validation, and test split. Training proceeded for 20 epochs using the Adam optimizer with a learning rate of 0.0001 and

CrossEntropyLoss, with early stopping monitoring validation accuracy.

The EfficientNetB0 Jupyter Notebook environtment was implemented in Keras/TensorFlow. Input images were resized to 224×224 pixels, and pretrained ImageNet weights were used as initialization. The training configuration employed a batch size of 32, a 75:25 training–validation split with no separate held-out test set, and 20 training epochs. The best-performing model checkpoint based on validation accuracy was saved automatically.

The InceptionResNetV2 Jupyter Notebook environtment used 150×150 input images with an 80:20 training–validation split and only two epochs of training. The classification head consisted of global average pooling, batch normalization, dense layers with ReLU activation, dropout, and a six-class softmax output. Approximately 955,014 parameters were set as trainable. Evaluation followed standard image classification metrics including accuracy, loss, confusion matrix, and per-class classification report [15].

TABLE 2. Dataset Distribution in the Local Folder

Model	Input	Main split	Epoch	Notes
Custom CNN	224×224	70:20:10	20	PyTorch; four conv blocks; global avg pool; early stopping
EfficientNetB0	224×224	75:25:00	20	Keras/TF; ImageNet init; best model checkpoint saved
MobileNetV2	224×224	75:25:00	20	Keras/TF; ImageNet init; inverted residual blocks
DenseNet121	224×224	75:25:00	20	Keras/TF; ImageNet init; dense block connectivity
InceptionResNetV2	150×150	80:20:00	2	Keras/TF; 955 K trainable params; large backbone

2.4 Mathematical Formulation

The main operation in a CNN is convolution between an input image or feature map and a kernel. For the output feature at position (i, j), the convolution operation can be expressed as:

$$Y_{i,j,k} = b_k + \sum_{c=1}^{C} \sum_{m=1}^{M} \sum_{n=1}^{N} X_{i+m,j+n,c}\ W_{m,n,c,k} \quad (1)$$

The non-linear activation function used in the CNN blocks is ReLU:

$$f(x) = \max(0, x) \quad (2)$$

The final softmax layer converts logits into class probabilities:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad (3)$$

The multiclass classification loss is categorical cross-entropy:

$$L = -\sum_{n=1}^{N} \sum_{i=1}^{K} y_{n,i} \log(p_{n,i}) \quad (4)$$

Model parameters are updated using Adam by estimating the first and second moments of the gradient:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

$$\theta_t = \theta_{t-1} - \alpha \frac{m_t}{\sqrt{v_t} + \epsilon} \quad (5)$$

The model is evaluated using accuracy, precision, recall, and F1-score:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (6)$$

2.5 Data Preprocessing and Augmentation

All images were resized to the input resolution required by each architecture. Pixel values were normalized to the range [0, 1] with per-channel mean and standard deviation standardization applied for PyTorch-based experiments and implicit Keras preprocessing applied for TensorFlow-based experiments. Batch normalization layers within the architectures also provide internal feature-level normalization, reducing sensitivity to input statistics.

Although the current experiments do not apply explicit offline augmentation beyond the default Keras ImageDataGenerator settings, the augmentation literature for tomato disease datasets consistently reports significant accuracy gains. GAN-based augmentation improved a tomato classification model from 95.3% to 97.6% in one recent study [16][17]. Standard augmentation strategies random horizontal and vertical flipping, rotation, brightness and contrast jitter, and random cropping are widely recommended as preprocessing baselines for agricultural image classification[18]. Future iterations of this work should incorporate online augmentation to improve generalization from curated datasets to real-field photographs.

III.RESULT AND DISCUSSION

3.1 Results

The custom CNN experiment progressed steadily from 26.57% training accuracy in epoch 1 to 68.79% in epoch 20. Corresponding validation accuracy improved from 35.14% to 70.97%. The final test evaluation produced an accuracy of 70.69% with a cross-entropy loss of 0.9497. This trajectory demonstrates that the architecture successfully learned discriminative features despite the absence of pretrained initialization, though the learning curve was relatively flat in later epochs, suggesting that the model's capacity was approaching its limit on this dataset.

EfficientNetB0 exhibited rapid early convergence, rising from 65.94% validation accuracy in epoch 1 to 89.44% by epoch 20, with a final validation loss of 0.3199. The learning curve was steep in the first five epochs and progressively stabilized, indicating that the pretrained feature representations required only moderate adaptation to the tomato leaf domain. This behaviour is consistent with transfer learning theory: ImageNet-pretrained models encode general visual primitives that substantially overlap with the texture and color gradients relevant to leaf disease classification.

MobileNetV2 attained 85.12% validation accuracy, demonstrating a strong trade-off between architectural complexity and classification performance. DenseNet121 reached 82.77% validation accuracy, reflecting the advantages of dense connectivity for preserving fine-grained lesion detail across feature map levels. These two results, along with EfficientNetB0, confirm that lightweight and moderately sized transfer learning models are well matched to datasets of approximately 7,000 images.

InceptionResNetV2 produced the lowest accuracy of 45.76% alongside a test loss of 1.4660, indicating that the model had not yet converged sufficiently to effectively classify the six-class tomato leaf disease dataset. The classification report from the corresponding Jupyter Notebook environment revealed strong prediction bias toward the early blight class, with several other classes assigned near zero recall. This pattern is characteristic of either severe underfitting attributable to the two epoch training schedule, or an evaluation pipeline issue such as label order mismatch between the trained model and the test data generator. The input resolution discrepancy (150 × 150 versus 224 × 224 for the other models) may also have degraded feature quality, particularly for lesion detail in smaller or elongated spots.

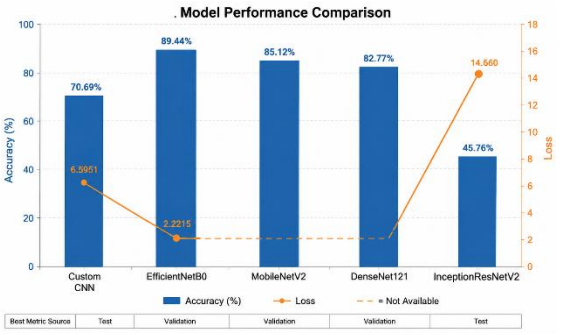

FIGURE 3. Model performance comparison based on the main Jupyter Notebook environtment metrics

TABLE 3. Summary of Experimental Results

Model	Best metric source	Loss	Accuracy
Custom CNN	Test	0,9497	70,69
EfficientNetB0	Validation	0,3199	89,44
MobileNetV2	Validation	–	85,12
DenseNet121	Validation	–	82.77

InceptionResNetV2	Test	1,4660	45.76

Table 3 presents a comparative summary of the experimental results obtained from five deeplearning models trained for tomato leaf disease classification across six disease categories.Each model was evaluated based on two primary metrics classification accuracy (%) and loss value recorded at the epoch that yielded the best observed performance within a training range of 20 to 50 epochs. It must be noted that the evaluation source is not uniform across all models: EfficientNetB0, MobileNetV2, and DenseNet121 are reported using validation-set performance, whereas the Custom CNN and InceptionResNetV2 are reported using test-set performance. This inconsistency arose from differences in the training monitoring strategy applied to each model and is acknowledged as a limitation of the current experimental setup. The "Best Metric Source" column explicitly documents the evaluation source for each model so that readers can interpret the reported figures accordingly.

Overall, these results demonstrate that transfer learning-based models, particularly EfficientNetB0, outperform both the custom baseline and the insufficiently trained InceptionResNetV2 under the current experimental configuration. The gap between EfficientNetB0 and the Custom CNN highlights the advantage of leveraging pretrained ImageNet weights for feature extraction in plant disease classification tasks. InceptionResNetV2, while architecturally capable, requires a longer training schedule or further hyperparameter tuning before it can be considered a viable candidate for this classification task. To ensure a fully rigorous comparison in future work, all models should be evaluated on a single, identical held out test set using a fixed random seed, and all loss and accuracy values should be recorded from that same evaluation source.

3.2 Discussion

The superiority of EfficientNetB0 over the other models in this study is theoretically grounded in the compound scaling methodology. Unlike depth-only scaling (ResNet) or width-only scaling (WideResNet), compound scaling simultaneously adjusts depth, width, and resolution under a unified coefficient, preserving the balance between receptive field size and feature map granularity. The MBConv blocks with squeeze-and-excitation recalibration further enable the model to selectively attend to spatially localized disease features lesion boundaries, chlorotic rings, and surface texture anomalies while suppressing uninformative background regions [19]. On a structured six-class tomato dataset where the discriminative signal resides predominantly in local textural and chromatic patterns, these architectural properties yield a natural advantage.

The 89.44% validation accuracy obtained by EfficientNetB0 is consistent with benchmarks reported in the recent literature. EfficientNet variants achieve between 97.3% and 99.5% accuracy on tomato leaf datasets of 3,000 to 10,000 images, with performance scaling monotonically with dataset size . The gap between those results and the present 89.44% is partially attributable to the smaller dataset (7,192 images vs. 10,000) and the absence of aggressive data augmentation in the current experiments. A fusion model combining EfficientNetB0 and MobileNetV2 for multi-crop disease detection achieved 89.5% accuracy on 22 classes, which aligns closely with the present single-architecture result for six tomato classes, suggesting that the current model's performance ceiling may be approximately consistent with what is achievable from this architecture on datasets of comparable scale.

The custom CNN's 70.69% test accuracy is a meaningful baseline given that it was trained from randomly initialized weights. Its per-epoch learning trajectory shows consistent but slow improvement, indicating that the four-block architecture successfully captures some disease-discriminative representations but lacks the depth and breadth to match transfer learning. Studies on rice, chili, and maize leaf diseases consistently show that custom CNNs without pretrained initialization plateau at 65–75% accuracy on datasets below 10,000 images, while transfer learning architectures routinely exceed 85% under comparable conditions [20]. To improve the custom CNN further, several strategies merit investigation: deeper architectures with residual connections to mitigate gradient dilution; learning rate scheduling such as cosine annealing; targeted augmentation including random erasing to simulate occlusion; and class-aware focal loss weighting.

The MobileNetV2 result of 85.12% confirms the architecture's viability for agricultural classification tasks at moderate dataset scales. Its depthwise separable convolutions efficiently capture spatial correlations in disease texture without the computational overhead of full convolutions, allowing rapid convergence within 20 epochs [13]. The DenseNet121 result of 82.77% is slightly lower, which may reflect the architecture's sensitivity to the number of dense blocks and transition compression rates. DenseNet's feature reuse strategy is beneficial for lesion-level classification because disease patterns at multiple scales of abstraction are preserved rather than discarded at transition layers, but the full benefit

may require more epochs for the dense connectivity patterns to stabilize [14].

The InceptionResNetV2 result necessitates a separate methodological discussion. With approximately 55.3 million parameters, InceptionResNetV2 is an order of magnitude larger than EfficientNetB0. Fine-tuning such a model for only two epochs on 5,753 training images (80% of 7,192) leaves the classification head severely undertrained: the model cannot adjust the multi-scale feature distributions of the Inception-Resnet stem to the color and texture characteristics of tomato disease imagery. This results in the classification head defaulting to majority-class predictions, producing the early blight bias observed in the Jupyter Notebook environtment report [5]. Comparative studies in maize and rice disease detection corroborate this finding, consistently showing that EfficientNet and MobileNet variants outperform InceptionResNetV2 under budget-constrained training regimes[15]. A fair evaluation of InceptionResNetV2 would require at minimum 20–30 epochs with progressive unfreezing, learning rate warm-up, and high-quality augmentation. From an application standpoint, EfficientNetB0 is the most deployment-ready candidate among the five architectures. Its 5.3 million parameters translate to a model file size below 25 MB after quantization, enabling deployment on Android devices with 2 GB of RAM. Inference times of less than 100 ms have been reported for EfficientNetB0 on commodity mobile hardware[6]. A study on oil palm fresh fruit ripeness classification using EfficientNetB0 with float16 quantization achieved 89.3% test accuracy with a 96 ms inference time per image, demonstrating the architecture's suitability for mobile agricultural applications [6], [7]. For tomato farmers in remote areas without stable internet connectivity, an on device model can provide immediate disease identification without latency introduced by cloud round trips.

Two methodological limitations must be addressed before the reported results can serve as a fully reliable benchmark. First, the current study evaluates EfficientNetB0, MobileNetV2, and DenseNet121 using validation-set performance, while the Custom CNN and InceptionResNetV2 are evaluated on the test set. This inconsistency makes direct cross-architecture comparison imprecise, as differences in reported accuracy may partially reflect test-validation split variation rather than genuine architectural advantages. A rigorous comparison requires all models to be re-evaluated on an identical held out test set with a fixed random seed. The authors acknowledge this as a priority for follow up experimentation and note that the current results should be interpreted with this caveat in mind. Second, overall accuracy alone is an insufficient summary metric for multi-class leaf disease classification. It is possible to achieve high accuracy while exhibiting near-zero recall on specific disease classes a model that systematically fails to detect septoria leaf spot or yellow curl virus would be clinically hazardous in a decision-support context. Macro-averaged F1-score, per-class precision-recall curves, and confusion matrix heatmaps should all be reported [12], [15].

IV.CONCLUSION

This study compared five deep learning architectures custom CNN, EfficientNetB0, MobileNetV2, DenseNet121, and InceptionResNetV2 for the classification of six tomato leaf disease categories using 7,192 images from a Kaggle dataset, with the objective of identifying which model best balances accuracy and efficiency for potential deployment in farmer-oriented diagnostic systems. EfficientNetB0 achieved the highest validation accuracy of 89.44% with the lowest validation loss of 0.3199 after 20 training epochs, attributable to its compound scaling methodology, squeeze-and-excitation channel recalibration, and effective leverage of ImageNet pretrained representations; MobileNetV2 and DenseNet121 followed at 85.12% and 82.77% validation accuracy, respectively, confirming that lightweight and moderately sized transfer learning models are well matched to agricultural datasets of this scale; theCustom CNN attained 70.69% test accuracy as a reasonable trained-from-scratch baseline; and InceptionResNetV2 produced only 45.76% test accuracy under a two-epoch configuration, demonstrating that large-parameter architectures require substantially longer training schedules to reach competitive performance on medium-sized, domain-specific datasets. It is important to note that the evaluation sources differ across models: EfficientNetB0, MobileNetV2, and DenseNet121 are reported on the validation set, whereas the Custom CNN and InceptionResNetV2 are reported on the test set. This inconsistency is a recognized limitation of the current study and should be resolved in future work by evaluating all models on a unified held-out test set. These findings directly address the research objectives by establishing that transfer learning consistently outperforms training from scratch on the given dataset size, that model complexity alone does not guarantee superior accuracy when training budget is constrained, and that EfficientNetB0 is the recommended architecture for mobile and edge deployment based on its accuracy efficiency trade-off. To realize the full potential of these results in practical agricultural environments, future work should standardize all model evaluations on a unified held out test set with macro F1-score and per-class confusion matrix reporting, validate models against real field tomato images captured under diverse lighting and occlusion conditions, and incorporate online data

augmentation and adaptive learning rate scheduling to close the performance gap with larger scale published benchmarks.

ACKNOWLEDGMENT

The authors would like to thank all parties who provided support and facilitated the completion of this research, including the Department of Information Technology at Politeknik Negeri Jember and the Department of Multimedia Technology at Akademi Komunitas Negeri Pacitan.

REFERENCE

[1] R. Soekarta, N. Nurdjan, and A. Syah, "Klasifikasi Penyakit Tanaman Tomat Menggunakan Metode Convolutional Neural Network (CNN)," *INSECT*, vol. 8, no. 2, 2023.

[2] R. A. Widarso, A. Sucipto, N. Rahma Yusrilfa Trisyayanti, H. Hasnira, T. Maharani, and D. R. Tisna, "Perancangan dan Implementasi Sirkulasi Air pada Rumah Kaca Hidroponik menggunakan Parameter Sensor Multivariabel berbasis Web," *Journal Of Applied Electrical Engineering*, vol. 9, no. 1, Jun. 2025.

[3] R. A. Kurniawan, A. Sunyoto, and A. Nasiri, "PENGARUH ARSITEKTUR CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI PENYAKIT DAUN TOMAT (EFFECT OF CONVOLUTIONAL NEURAL NETWORK ARCHITECTURE FOR TOMATO LEAF DISEASE CLASSIFICATION)," *Teknimedia:Teknologi Informasi dan Multimedia*, vol. 4, no. 2, 2023, [Online]. Available: https://www.kaggle.com/datasets/emmarex/plantdiseas

[4] Tsabit yunan al rajab and N. Nafiiyah, "Evaluasi Kinerja Model CNN Berbasis Transfer Learning dalam Klasifikasi Penyakit Daun Padi," *Jurnal Sistem Komputer dan Informatika (JSON)*, vol. 7, no. 3, pp. 989–995, Mar. 2026, doi: 10.30865/json.v7i3.9539.

[5] A. C. Milano, "KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN MODEL DEEP LEARNING EFFICIENTNET-B6," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 1, Jan. 2024, doi: 10.23960/jitet.v12i1.3855.

[6] D. Putri and A. Dkk, "AUGMENTASI DATA PADA IMPLEMENTASI CONVOLUTIONAL NEURAL NETWORK ARSITEKTUR EFFICIENTNET-B3 UNTUK KLASIFIKASI PENYAKIT DAUN PADI," *ZONAsi: Jurnal Sistem Informasi*, vol. 5, no. 2, pp. 239–249, May 2023.

[7] K. A. Tamayasa and L. J. E. Dewi, "ANALISIS PERBANDINGAN MODEL ARSITEKTUR MOBILENETV2 DAN EFFICIENTNETB3 DALAM KLASIFIKASI PENYAKIT DAUN JAGUNG," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 14, no. 1, Jan. 2026, doi: 10.23960/jitet.v14i1.8624.

[8] F. Alfarisi, "DIAGNOSA HAMA PADA TANAMAN PADI MENGGUNAKAN METODE ARTIFICIAL NEURAL NETWORKS (ANN)," Malang, Jun. 2023.

[9] W. Fithratul Zalmi, H. Saputro, J. Sitanggang, and K. Leatemia, "PENERAPAN CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK KLASIFIKASI PENYAKIT DAUN TOMAT," *IFTK : Jurnal Informatik*, 2025.

[10] S. Anam Alidrus, M. Aziz, O. Virgantara Putra, and U. Darussalam Gontor Jl Raya Siman KecSiman KabPonorogo, *Deteksi Penyakit Pada Daun Tanaman Padi Menggunakan Metode Convolutional Neural Network*. 2021.

[11] S. Azizah, A. I. Pradana, and D. Hartanti, "Identifikasi Kesehatan Daun Tanaman Padi Menggunakan Klasifikasi Biner Sehat dan Tidak Sehat dengan Algoritma Convolutional Neural Network (CNN) Di Kabupaten Klaten," *Komputika : Jurnal Sistem Komputer*, vol. 13, no. 2, pp. 173–181, Oct. 2024, doi: 10.34010/komputika.v13i2.12771.

[12] E. F. P. - and B. I. -, "KLASIFIKASI PENYAKIT TANAMAN PADI MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK DENGAN ARSITEKTUR MOBILNETV2," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 14, no. 1, Jan. 2026, doi: 10.23960/jitet.v14i1.8900.

[13] F. N. Firdaus, "Analisis Kinerja MobileNetV2 dalam Klasifikasi Penyakit Daun Cabai Berdasarkan Citra," in *SEMINAR NASIONAL TEKNOLOGI DAN SAINS*, Universitas Nusantara PGRI Kediri, 2026.

[14] E. Hassan, S. A. Ghazalah, N. El-Rashidy, T. A. El-Hafeez, and M. Y. Shams, "DenseNet Model with Attention Mechanisms for Robust Date Fruit Image Classification," *International Journal of Computational Intelligence Systems*, vol. 18, no. 1, Dec. 2025, doi: 10.1007/s44196-025-00809-4.

[15] M. Alruily *et al.*, "Ensemble deep learning for Alzheimer's disease diagnosis using MRI: Integrating features from VGG16, MobileNet, and InceptionResNetV2 models," *PLoS One*, vol. 20, no. 4 April, Apr. 2025, doi: 10.1371/journal.pone.0318620.

[16] M. Ulfi, M. I. A. Supriyanto, C. Mala Sari, and S. A. Nuswantoro, "IDENTIFIKASI PENYAKIT PADA TANAMAN PADI MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN)," *Jurnal Sains Komputer dan Teknologi Informasi*, vol. 7, no. 2, May 2025.

[17] Sintia, N. Arifin, and C. N. Insani, "Implementasi Mobilenetv2 Pada Klasifikasi Penyakit Daun Bibit Kakao Berbasis Pengolahan Citra Digital," *INSECT: Informatics and Security, Jurnal Teknik Informatika*, vol. 12, no. 1, 2026.

[18] A. P. Aryaputra, D. W. Widodo, and A. Sanjaya, "Klasifikasi Penyakit Daun Tomat Menggunakan Algoritma CNN Mobilenet V2," in *SEMNAS INOTEK(Seminar Nasional Inovasi Teknologi)*, Online, 2025, pp. 2549–7952.

[19] M. Wahyuni, "Klasifikasi Penyakit Daun Tomat dengan Perbandingan Fungsi Aktivasi Multi Layer Perceptron," *Jurnal Minfo Polgan*, vol. 13, no. 2, pp. 1988–1998, Dec. 2024, doi: 10.33395/jmp.v13i2.14351.

[20] F. Handayani, B. Baidarus, S. Sunanto, B. A. Putra, C. Anggraini, and R. M. Taufiq, "Klasifikasi Citra Penyakit Daun Tomat Menggunakan Metode Convolutional Neural Network (CNN) Dengan Arsitektur VGG-19," *Jurnal CoSciTech (Computer Science and Information Technology)*, vol. 6, no. 3, pp. 405–413, Dec. 2025, doi: 10.37859/coscitech.v6i3.10699.

RIFQI AJI WIDARSO is a lecturer and researcher in the field of Information Technology with expertise in computer networks, software engineering, web development, and artificial intelligence applications. He earned his Bachelor of Engineering (S.T.) and Master of Engineering (M.T.) degrees in Information Technology-related disciplines and has been actively involved in academic teaching, scientific research, and community service activities.His research interests include deep learning, Internet of Things (IoT), image processing, decision support systems, and intelligent monitoring systems for agriculture and smart environments. In addition to his academic activities, he is also experienced in desktop and web-based application development, particularly using Java and Laravel frameworks.As an academic, Rifqi Aji Widarso has contributed to various research publications, conference proceedings, and student mentoring activities in the fields of information systems and computer science. He is committed to

developing innovative technology solutions that provide practical benefits for education, industry, and society.

ADI SUCIPTO is an academic and practitioner in the field of Information Technology with a strong focus on applied computing, software development, and digital technology implementation. He obtained his Diploma IV (S.ST.) and Master of Applied Technology (M.Tr.) degrees in technology-related disciplines and has extensive experience in teaching, research, and applied technological innovation..

DHONY MANGGALA PUTRA is an academic and researcher in the field of Information Technology with interests in software engineering, intelligent systems, and digital innovation. He has been actively involved in teaching, research, and technology development activities, particularly in the areas of computer science and information systems.

TAMARA MAHARANI is a researcher and technology enthusiast with interests in information technology, digital systems, and intelligent computing applications. She has been actively involved in academic and research activities related to software development, data analysis, and emerging technologies..